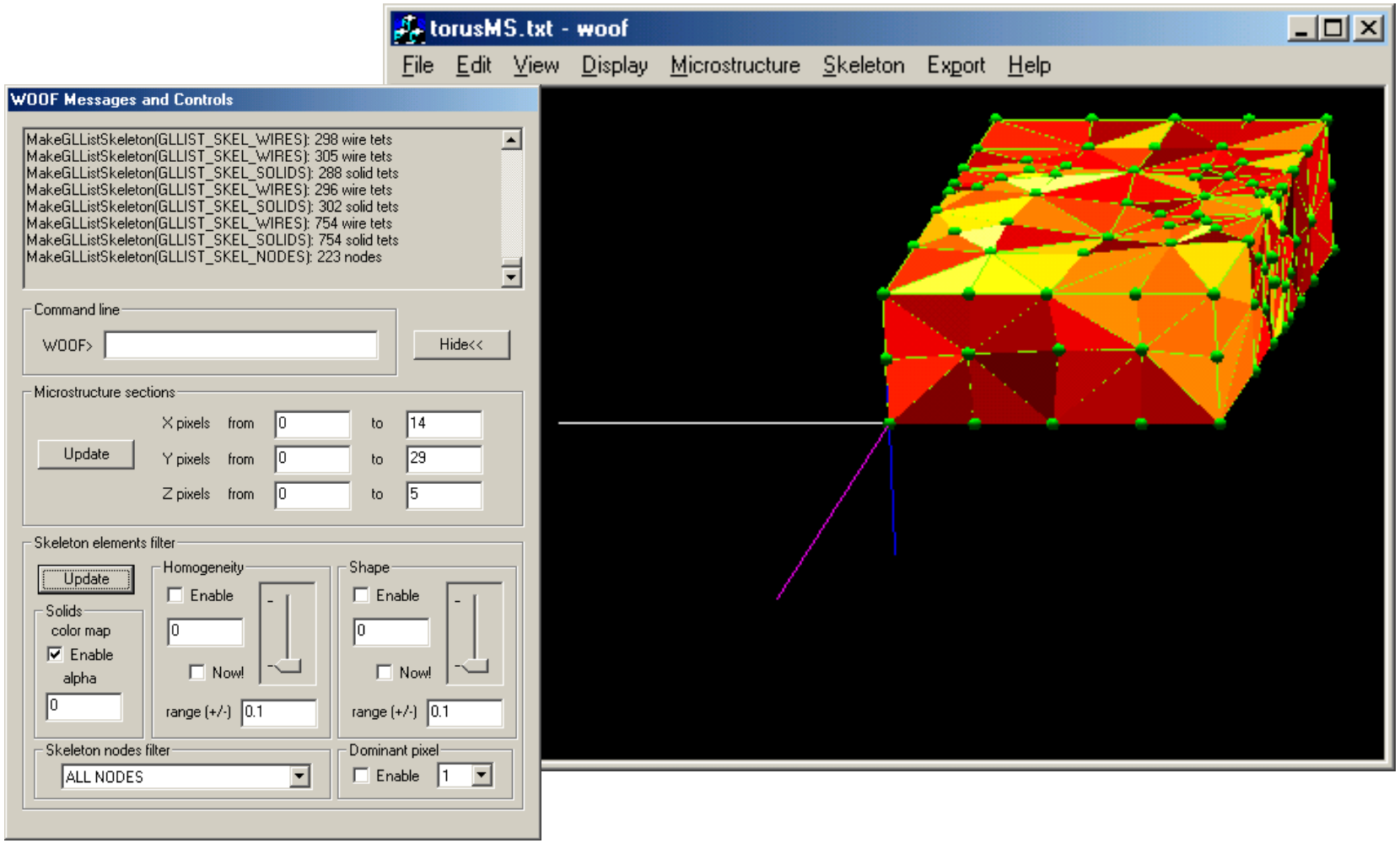
OOF Project Update

Rhonald Lua

Outline

- Program for performing an OOF-style construction of a skeleton/mesh over a 3-D microstructure as scaffold
- Status of parallel/distributed OOF2
- Status of OOF on Windows

OOF2-style meshing in 3-D

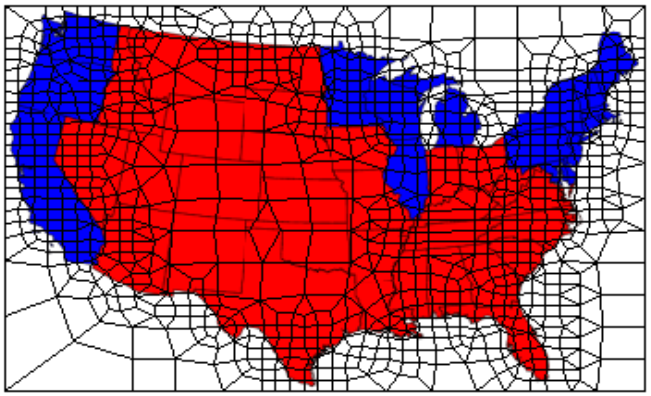


Goals

- Write a 3-D skeleton-modification engine, mimicking that in OOF2, that may form a basis of study and perhaps incorporated in a future version of OOF2 (i.e. OOF3).
- Build functioning mock-applications on top of the skeleton-modification routines to explore the feasibility of an OOF3.

Recall what OOF2 does

OOF2 fits an FE mesh on top of a microstructure derived from a micrograph or image:



http://www.ctcms.nist.gov/~rlua/oof2
Red-States/Blue-States image taken from
Gastner, Shalizi, Newman

...such that each mesh element:

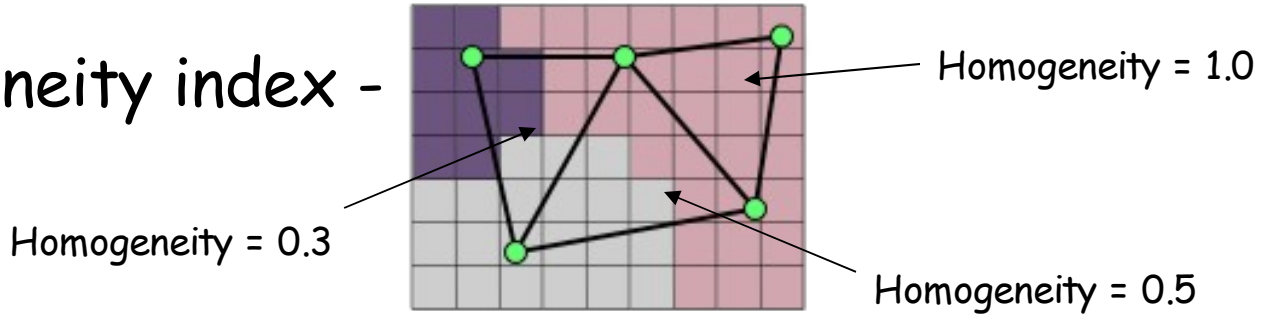
1. has a 'good' shape,
2. encloses a single type of material or color (i.e. homogeneous),
3. is much larger than a single pixel.

OOF terminology primer

1. Skeleton – defines the geometry of the mesh.

   Skeleton-modification – tools for modifying a skeleton so that it follows material boundaries.

2. Homogeneity index -

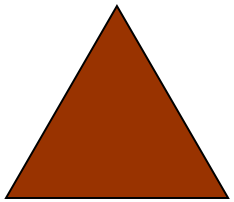


(http://www.ctcms.nist.gov/~langer/oof2man/Section-Concepts-Skeleton.html)

Homogeneity energy: E_h = 1.0 – (Homogeneity index)

(continued) OOF terminology

3. Shape index -



Shape index = 1.0

Shape index close to 0

Shape energy: $E_s = 1.0 - (\text{Shape index})$

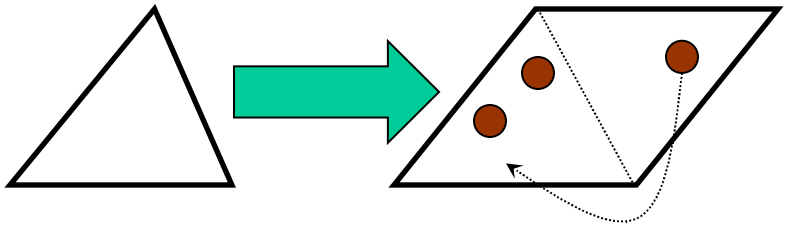
4. Effective element energy: $E = \alpha E_h + (1-\alpha)E_s$

E is a measure of the desirability of an element.
It is used as an energy in skeleton annealing.
Low E is generally good. Parameter 'alpha' (α) tunes the emphasis between homogeneity and shape.

Calculation of indices in OOF3 mock-up

1. Homogeneity index – statistically sample the interior of the element for the dominant pixel.

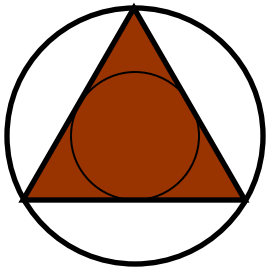
Example: get a set of random points inside triangle.



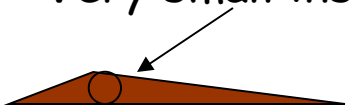
Generate random points within parallelogram. Fold back points that fall outside the triangle

http://vcg.iei.pi.cnr.it/activities/geometryegraphics/pointintetraedro.html

2. Shape index – Ratio of the size of the inscribing sphere to the circumscribing sphere (times a normalizing factor).



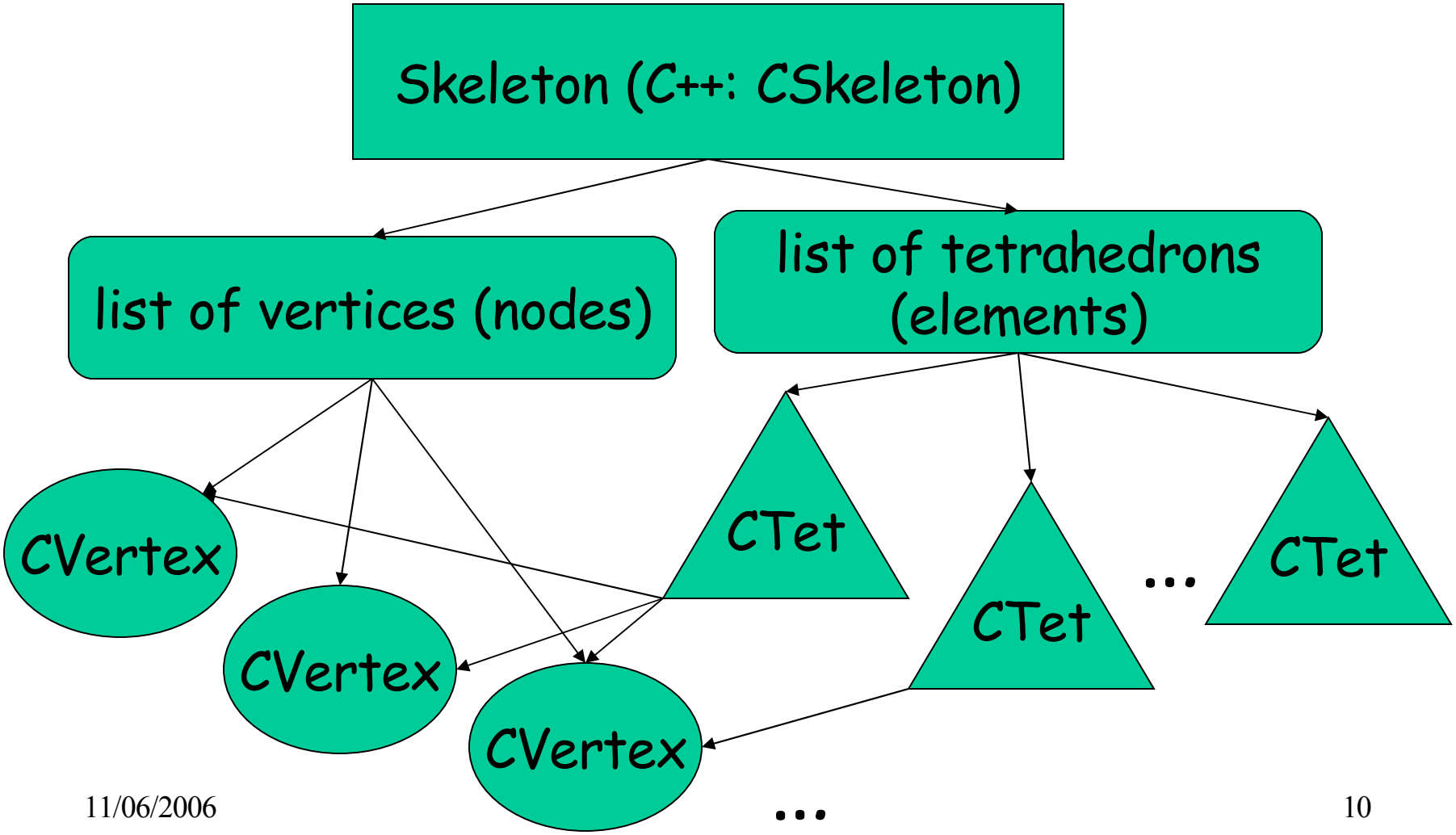
Very small inscribed sphere!



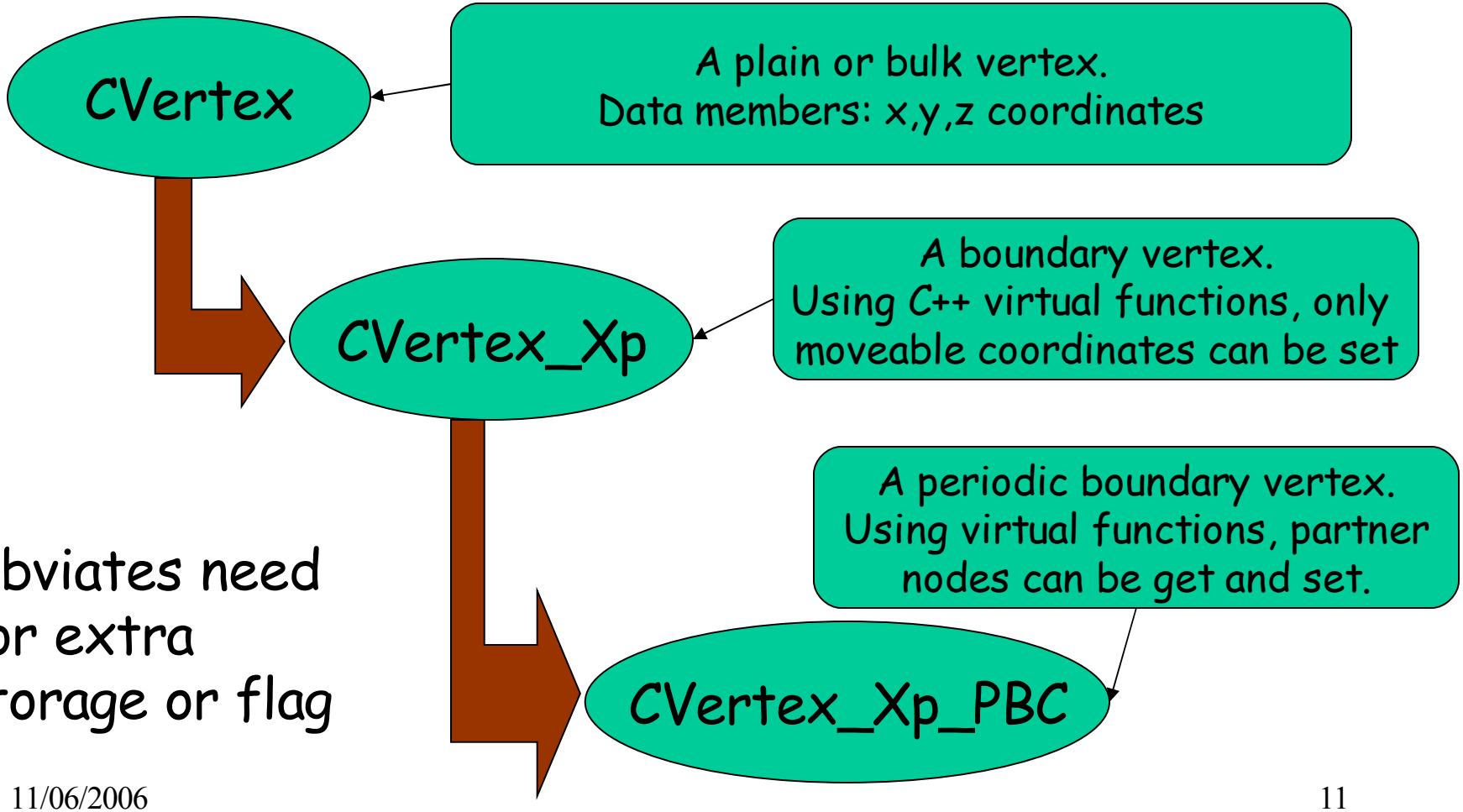
Design philosophy for OOF3 mock-up

- Extend the skeleton-modification methods of OOF2 into 3-D.
- Use as little run-time computer memory as possible (Memory that scales with the problem size). If you can compute it, then don't store it.
- Simple, easy-to-use programming interface.
- Provide option to make skeleton compatible with periodic boundary conditions.

Skeleton data structure



Class hierarchy for skeleton vertex



Obviates need for extra storage or flag

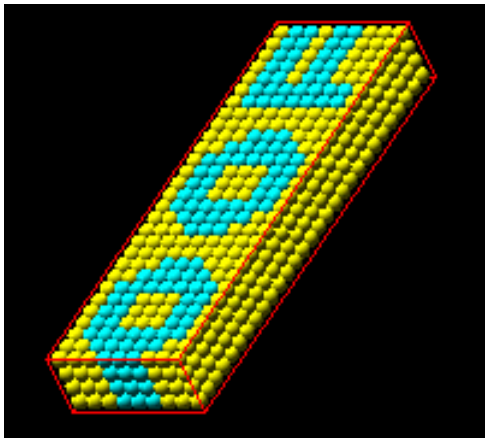
Microstructure format

Simple two-pixel color 3-D array.
(Sufficient to define a machine part)

Input file

```
#First line must be pixel dimensions nx,ny,nz
9 29 4
#Second line must be physical dimensions sx,sy,sz
9.0 29.0 4.0
#Pixel data
000000000000000000000000000000
001111100000011111100001111111
011111111000011111111000111111
111000111001110001110011000000
111000111001110001110011111111
111000111001110001110011111111
011111111000011111111000100000
001111100000011111100001100000
000000000000000000000000000000
#
000000000000000000000000000000
001111100000011111100001111111
```

Graphics output



Use 1-D array of chars or bytes for storage: 00111110,00…

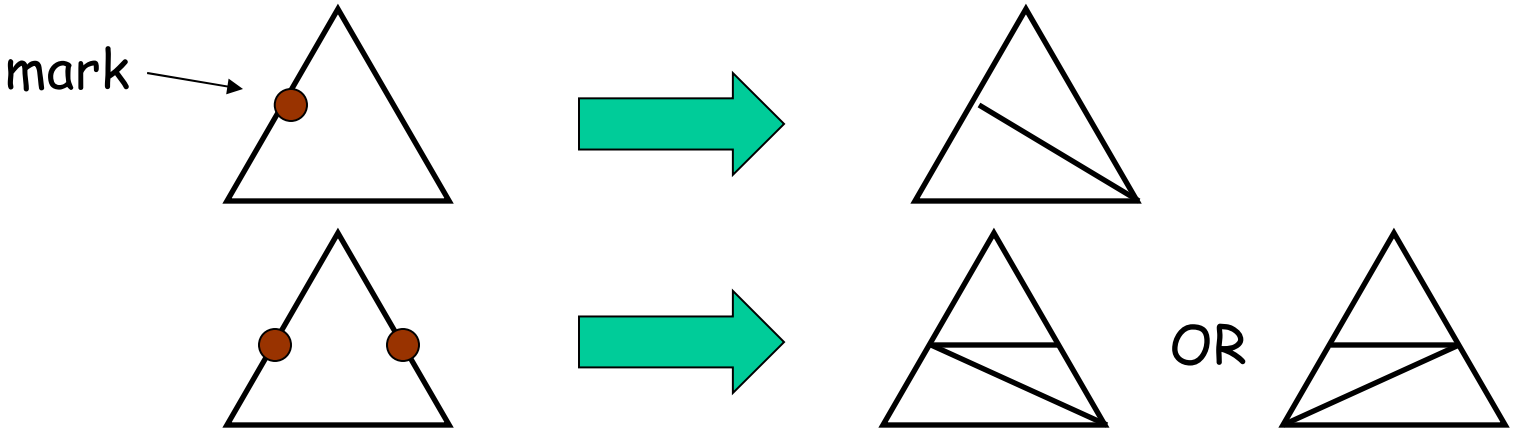
Each bit encodes a single pixel or phase

Skeleton-modification techniques

- **Refinement –** break down tetrahedral elements into smaller tetrahedrons.
- Simulated annealing:
    1. Random (gaussian) nodal displacements.
    2. Nodal motion toward average neighbors' position.
    3. Nodal motion along an element edge towards microstructure interfaces (transition points).

Refinement

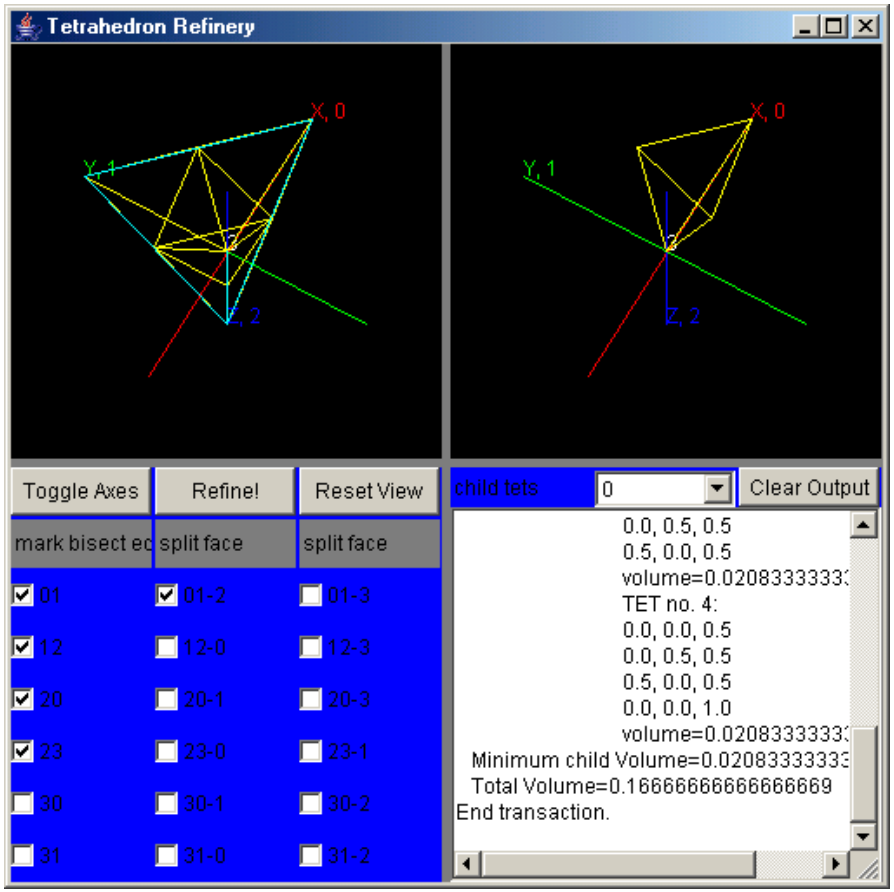
Mark one or more element edges for bisection depending on element homogeneity or other user preference. Examples:



Edge bisection will induce the bisection of the element faces as well as those of neighboring elements.

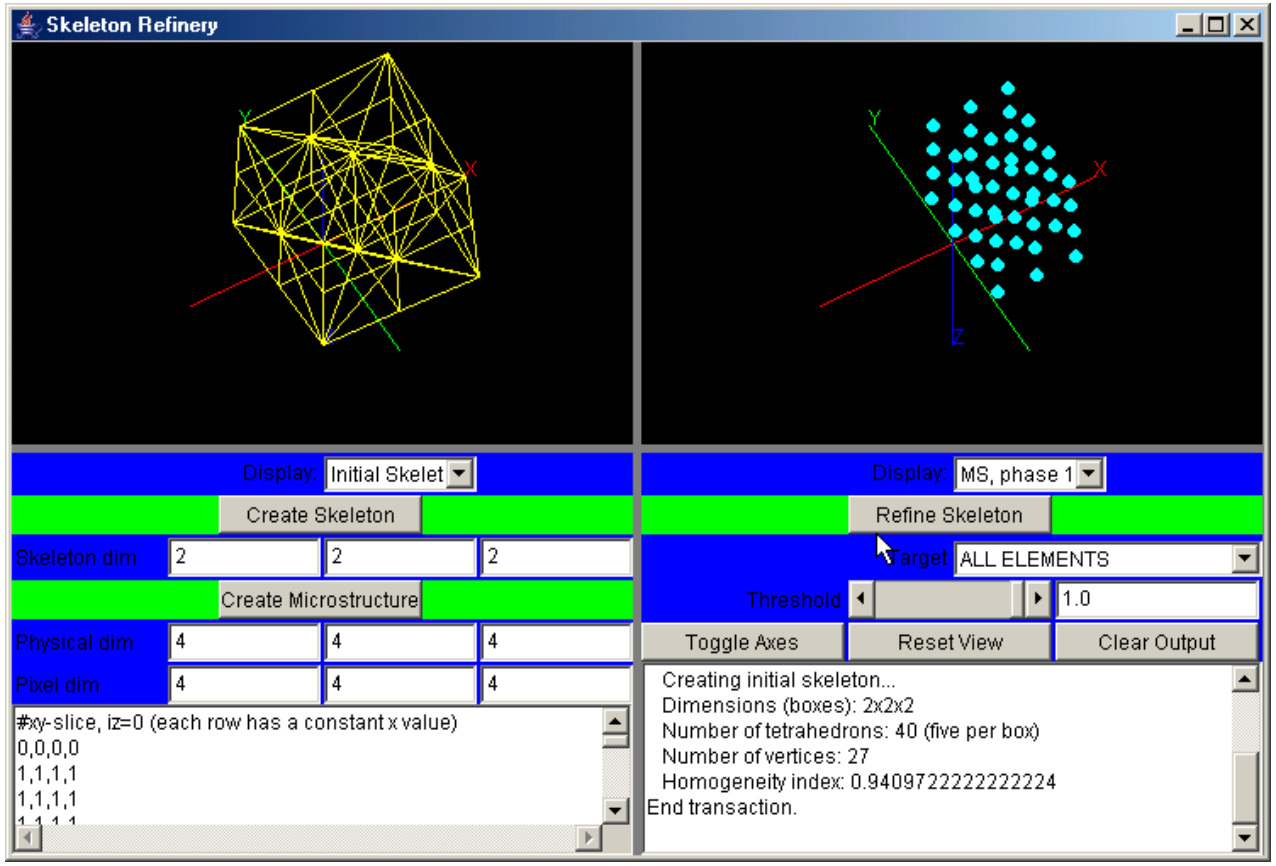
Tetrahedron Refinement: Java demo

(http://www.ctcms.nist.gov/~rlua/FE/refinery)



Skeleton Refinement: Java demo

(http://www.ctcms.nist.gov/~rlua/FE/refinery/skeleton)



Annealing

Monte Carlo molecular dynamics of the nodes

Step 1: Pick nodes or vertices to move.

Step 2: Perform a trial move of each node.

Move possibilities:

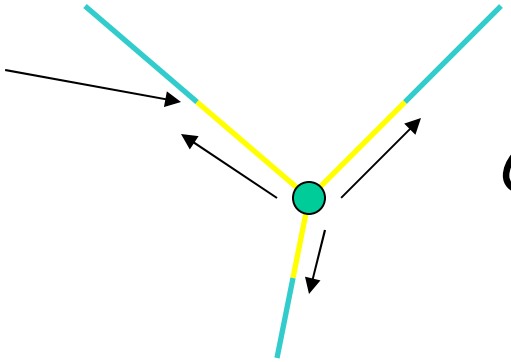
1. Standard - random (gaussian) nodal displacements
2. Smoothing – move node towards the average position of its neighbors.
3. Snap-nodes – move node along an edge towards the interface between two phases, called transition points.

Step 3: Accept or reject the move based on a boltzmann factor: $\exp(-\Delta E/T) > r$

r – random number in [0,1]

(continued) Annealing: Snap-nodes

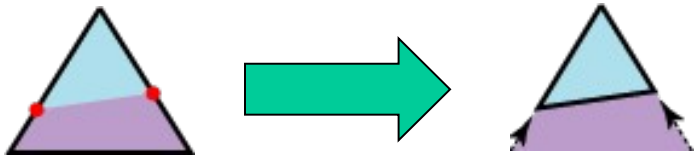
Transition points



Choose the best move

Correlated Snap-nodes:

Example:



(http://www.ctcms.nist.gov/~langer/oof2man/RegisteredClass-SnapNodes.html)

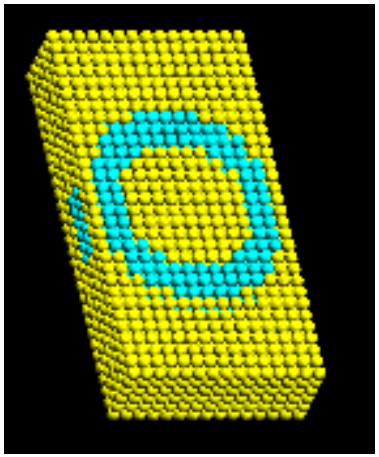
(Present in OOF2, but not implemented yet in mock-ups.)

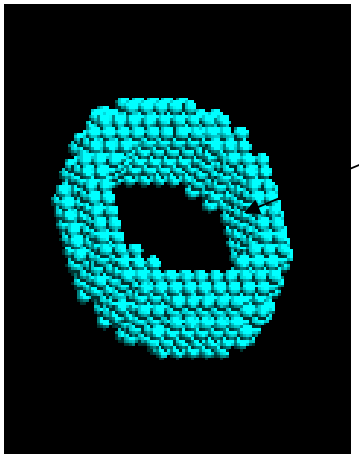
OOF3 Demo

WOOF – a C++/MFC/OpenGL application

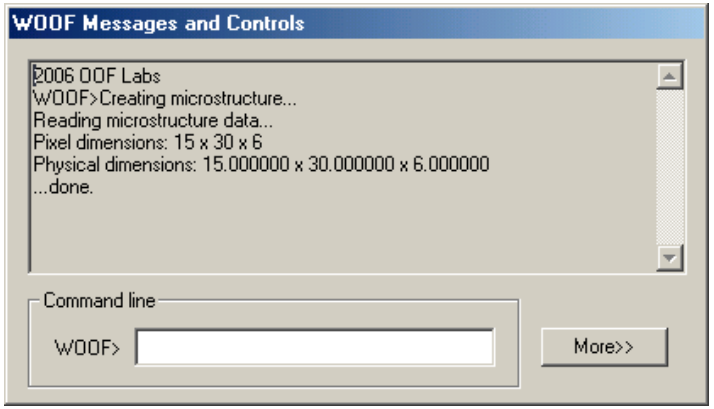
Create a microstructure:

Phase-0 (yellow)
Phase-1 (cyan)

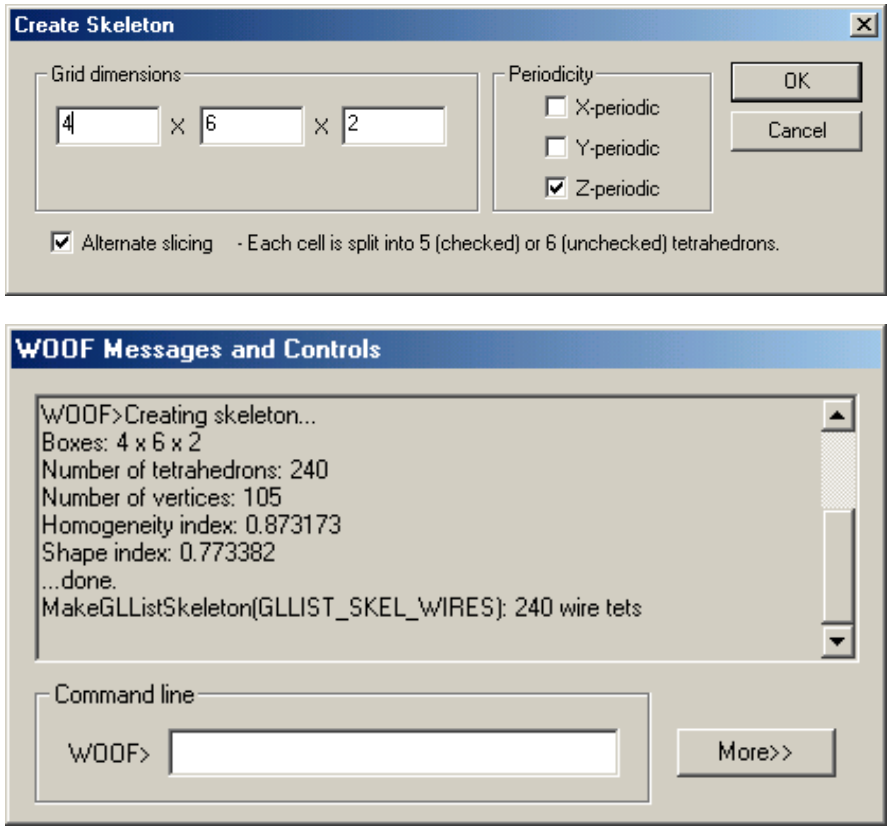


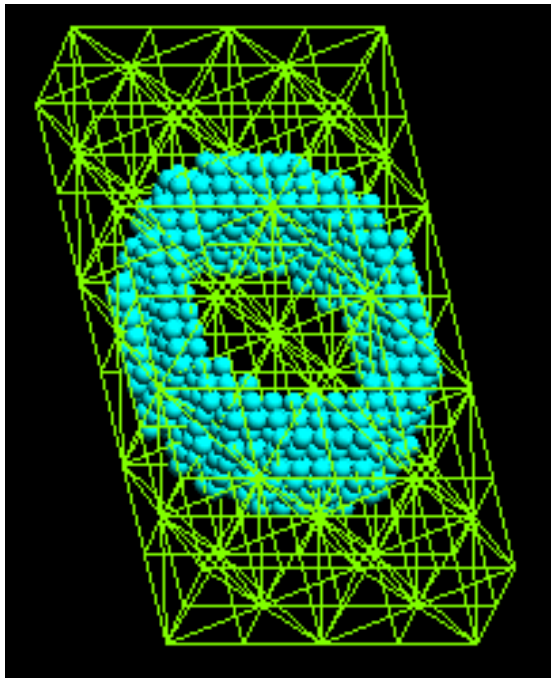


Phase-1 forms a donut (torus)



(demo continued) Create an initial skeleton

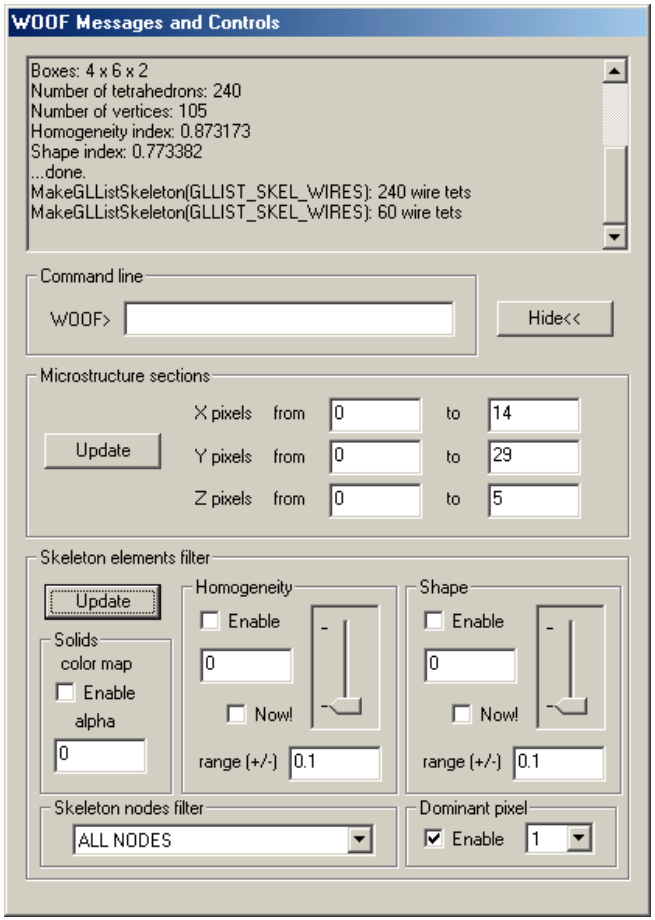


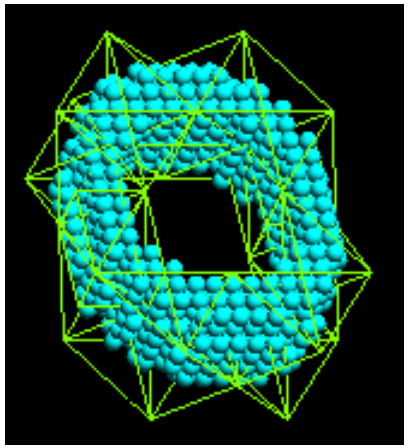


Because of z-periodic boundary condition, mesh topology is also a torus! (preserved after any skeleton modification)

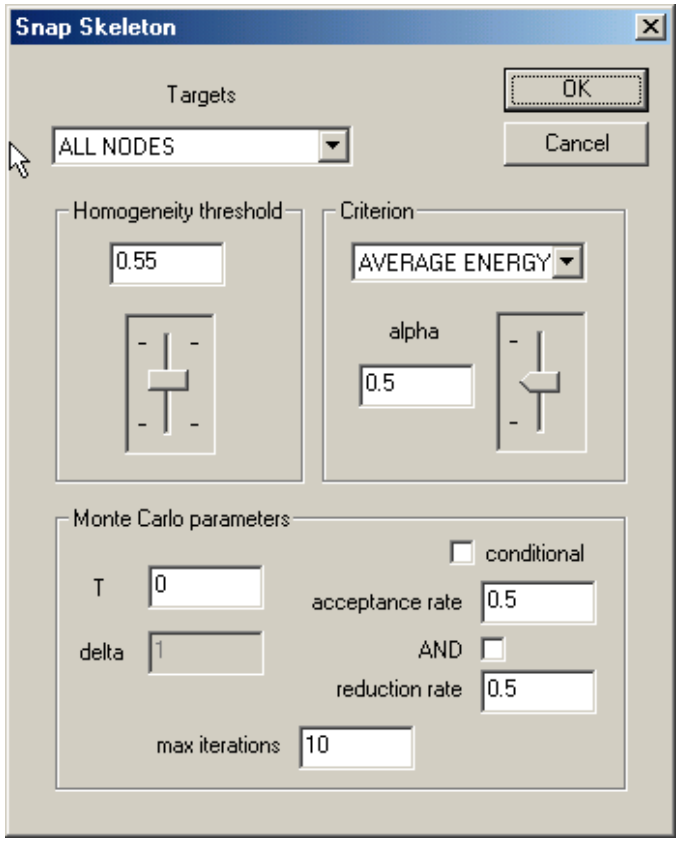
(demo continued)

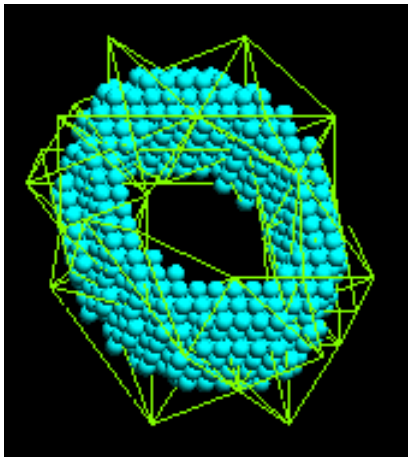
Display wire-frame of elements that contain phase-1 as a dominant pixel



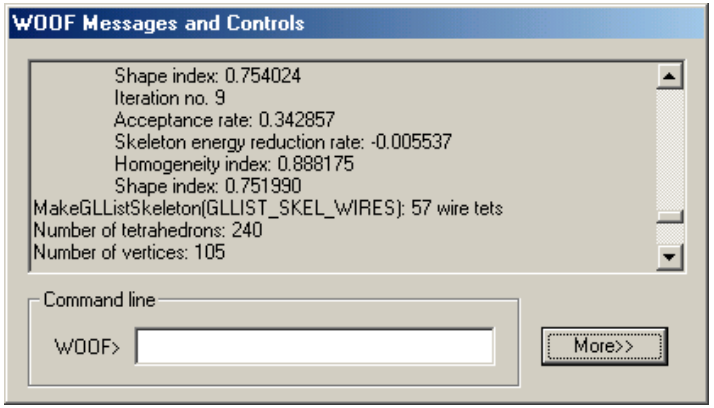


(demo continued)Snap-nodes of initial skeleton



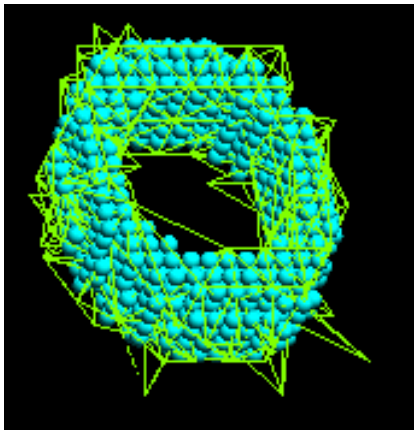


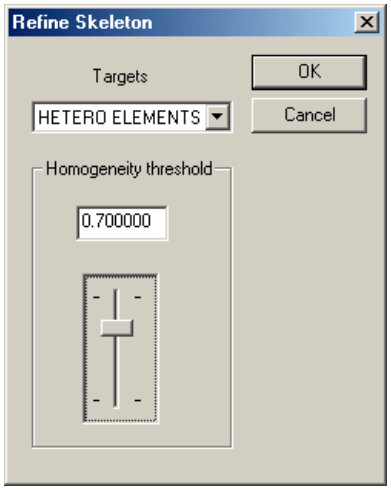
Homogeneity index increased by 0.01 and shape index decreased by 0.02. Skeleton changed little.

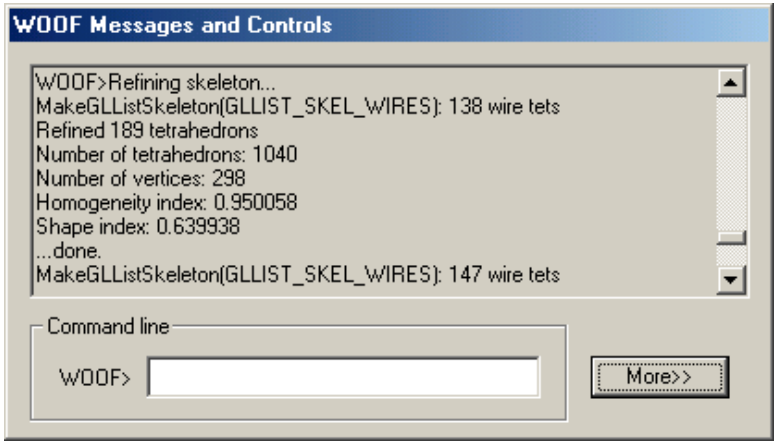


(demo continued) Refine heterogeneous elements in skeleton

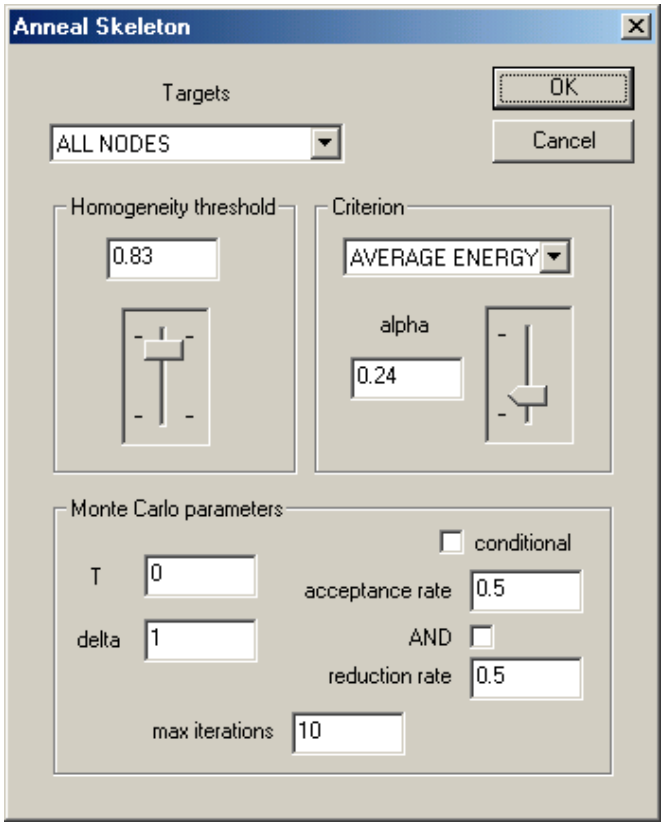
Homogeneity index increased considerably (up 0.06) but shape index went down (-0.11). Let us use anneal and smooth to try to remove the sharp edges that can injure an engineer!

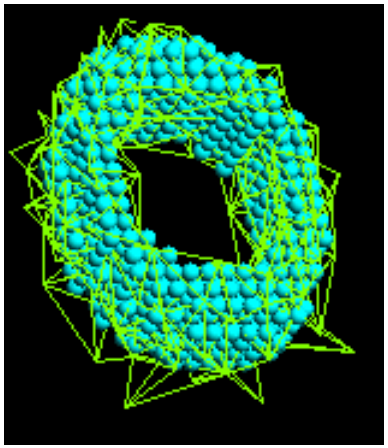




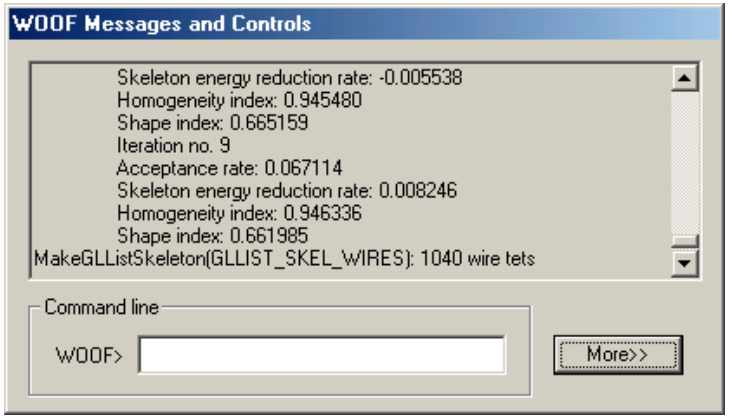


(demo continued) Standard Anneal

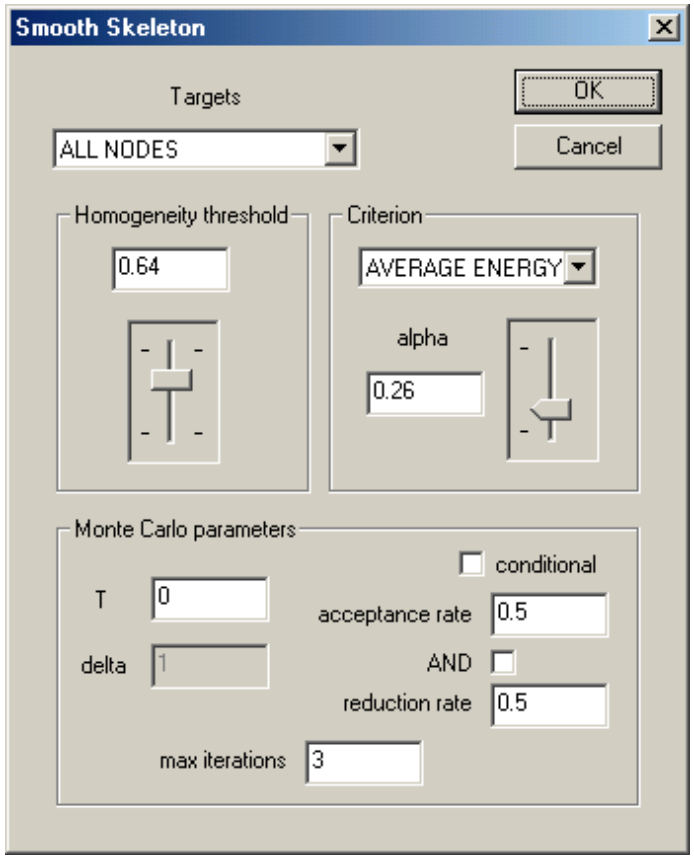


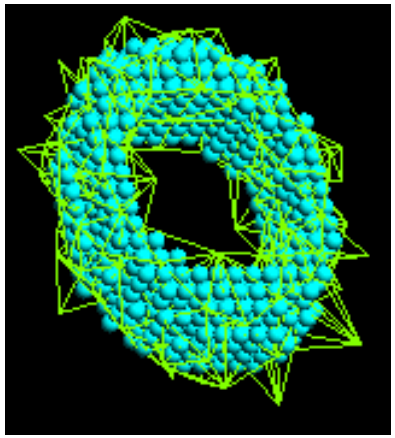


A modest change, shape index went up by 0.02.

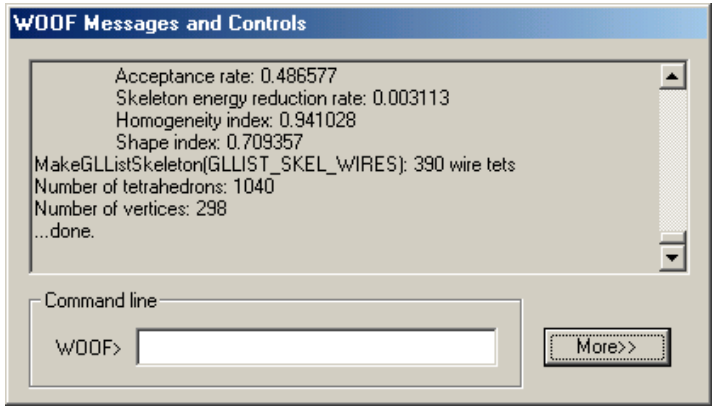


(demo continued) Smoothing

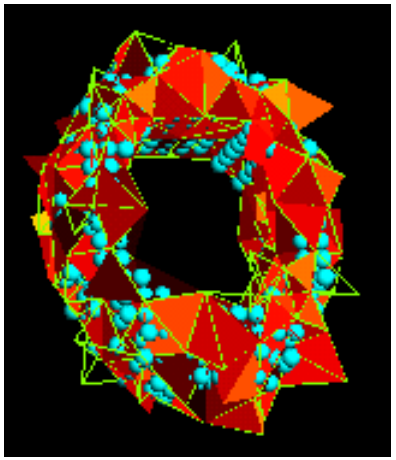




Again, a modest change, shape index went up by 0.04.

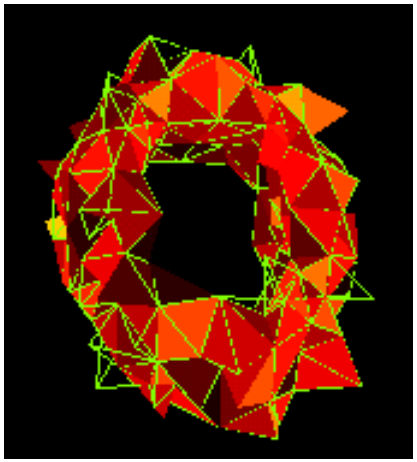


(demo continued) Color-map of skeleton solids indicating element energy



microstructure (phase-1) + wires + solids

Cool colors (white,yellow) indicate badly-shaped elements

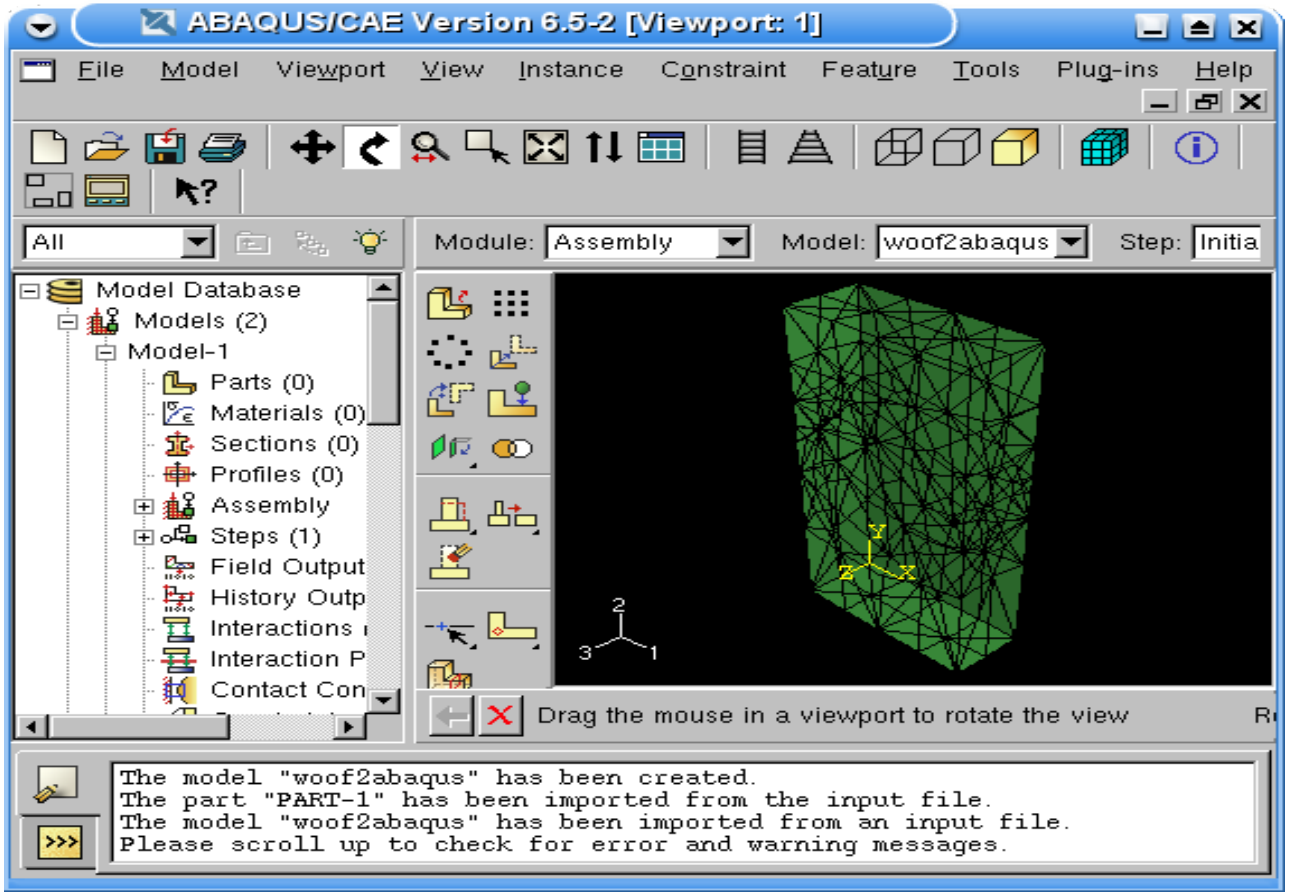


wires + solids

Verifying the correctness of the skeleton

- Test that the tetrahedrons are non-singular, and that $\Sigma V_{tets} = V_{box}$.
- Congruence of skeleton - Check that a face of a tetrahedron matches with exactly one other face in another element.
- Export the skeleton into abaqus (cae)

Exported skeleton: OOF3 to Abaqus



Caveat!

This is a mock-up (albeit a working mock-up).

Features in OOF2 that are missing in this mock-up:

- Scriptability – be able to save/record your work and 'play' them back later. Key to automation.
- Defining materials and properties
- Image processing and skeleton manipulation
- FEM solver
- Parallel/distributed computing capability
- Portability – e.g. for GUI interface code.
- Working with multiple microstructures, skeletons or mesh at the same time. Undo/Redo mechanism.
- And the list goes on…

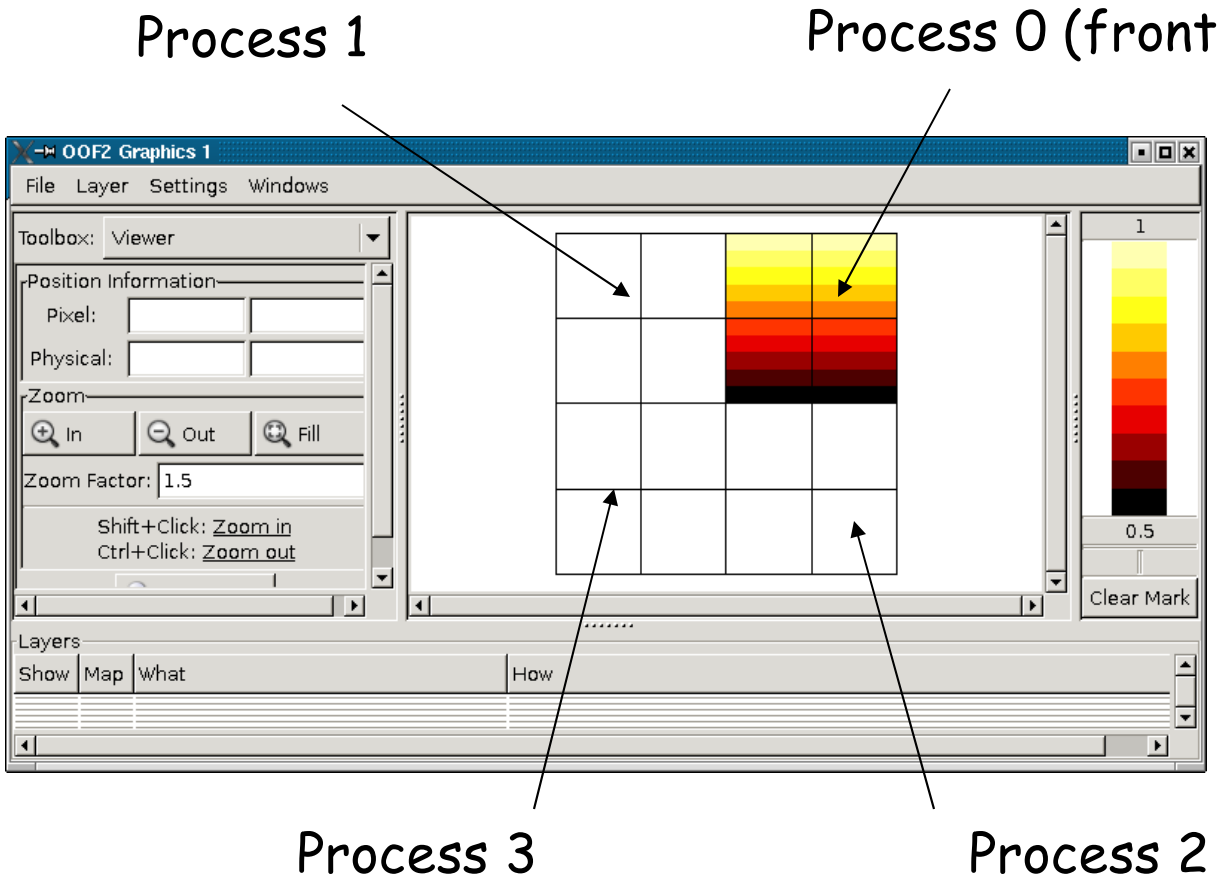
Many of these features will carry over to OOF3 when OOF2 is used as a code-base.

Options for Graphics and GUI toolkits

- Java AWT (Abstract Window Toolkit)
- VPython
- OpenGL + GTK. (Trivia: The dinosaurs in *Jurassic Park* and the T1000 in *T2* were OpenGL applications.)
- VTK (Visualization ToolKit)
- OpenDX

(Shifting gears…)

Parallel/Distributed mode of OOF2

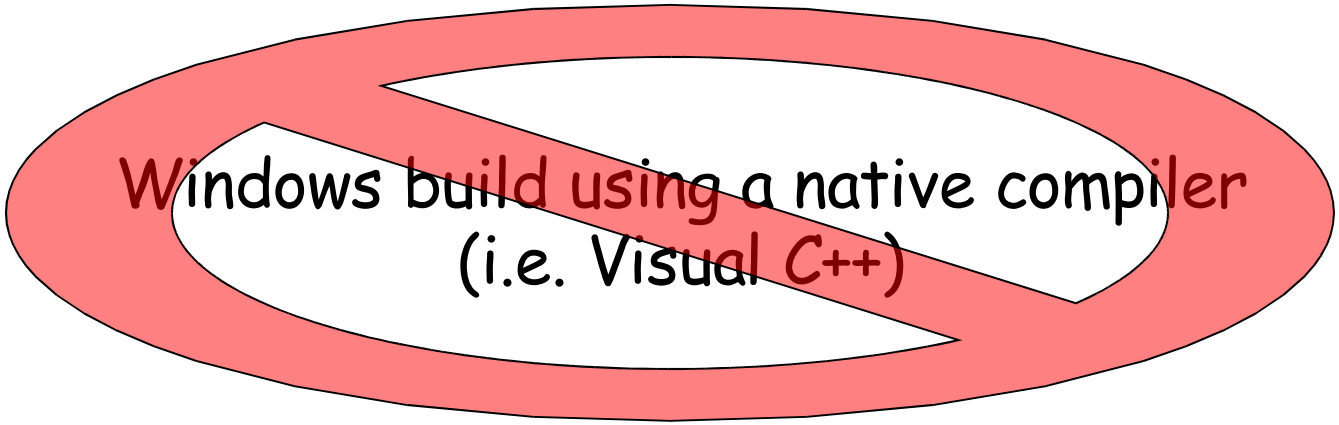


(continued) Parallel OOF2

TODO:

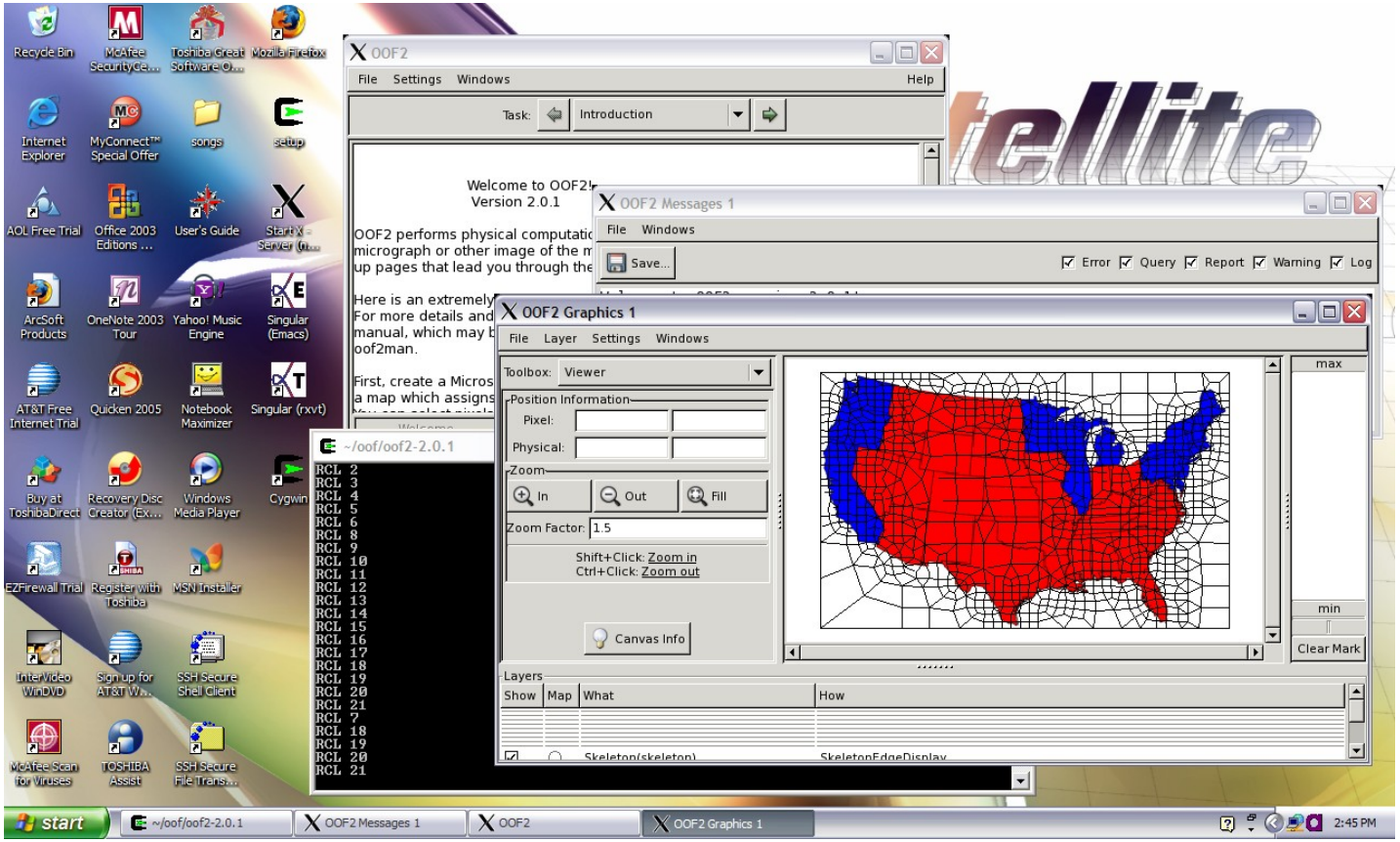
- Combine the data and results into a single graphics output at the front-end.
- Rigorous testing in a cluster. Professor R. Edwin Garcia has offered his computing resources at prometheus.ecn.purdue.edu (A Mac cluster).

OOF on Windows



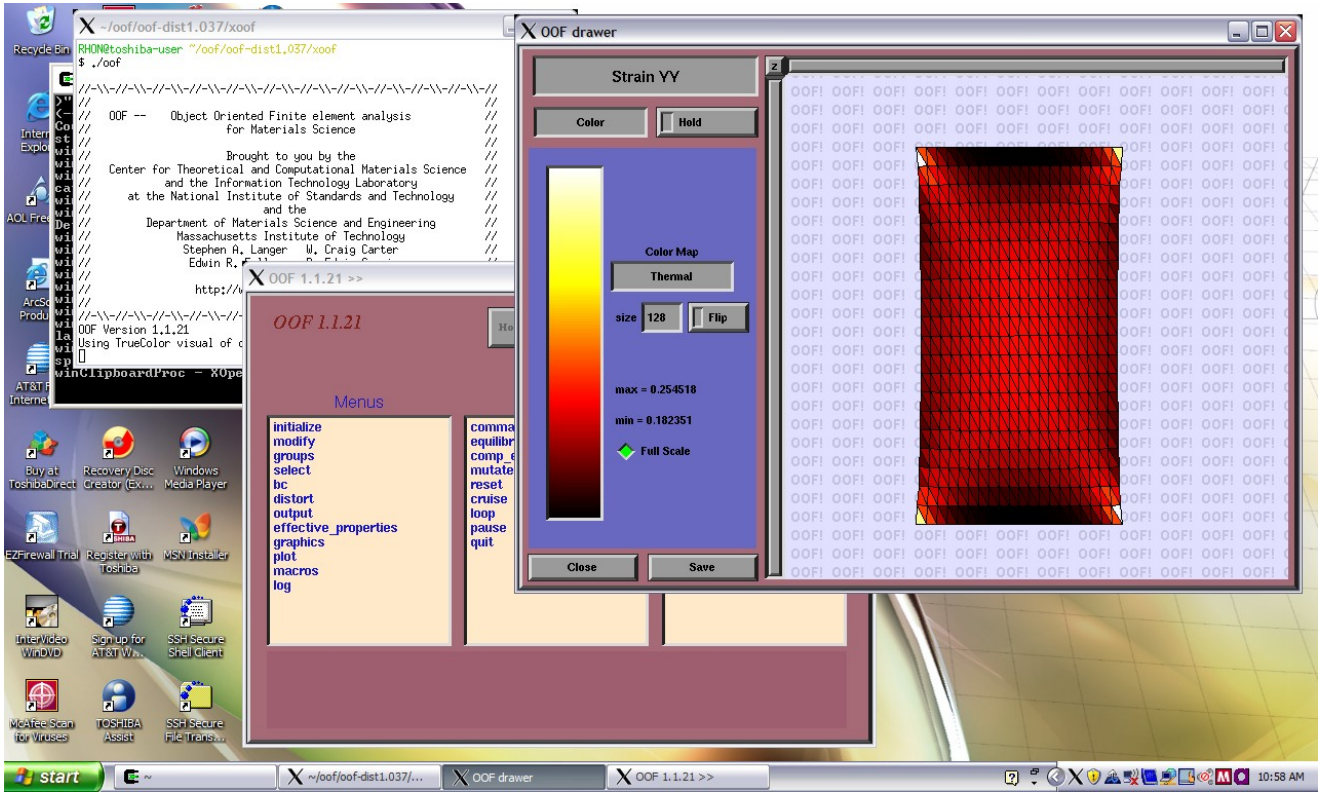
Instead employ a Cygwin solution. Cygwin (www.cygwin.com) is a tool for building Unix programs that will run on Windows.

OOF2 on Cygwin



http://www.ctcms.nist.gov/~rlua/FE/win/oof2.html

OOF1 on Cygwin



http://www.ctcms.nist.gov/~rlua/FE/win/oof1.html

Acknowledgments

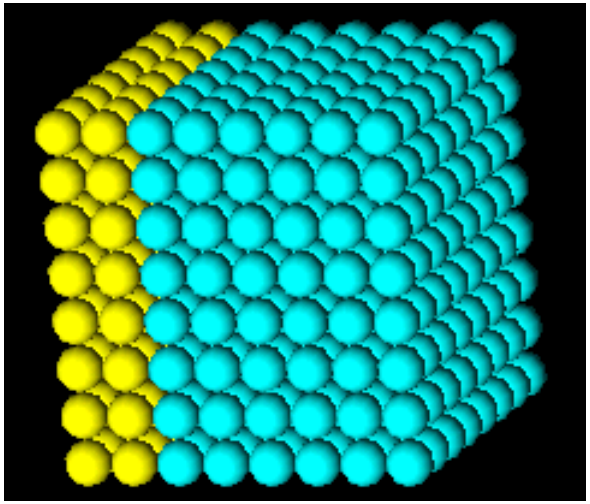
- Steve Langer, Andrew Reid (OOF mentors)
- Seung-Ill Haan (R. Lua learned the style of refining a tetrahedron from him.)

Visit: http://www.ctcms.nist.gov/~rlua/matcase

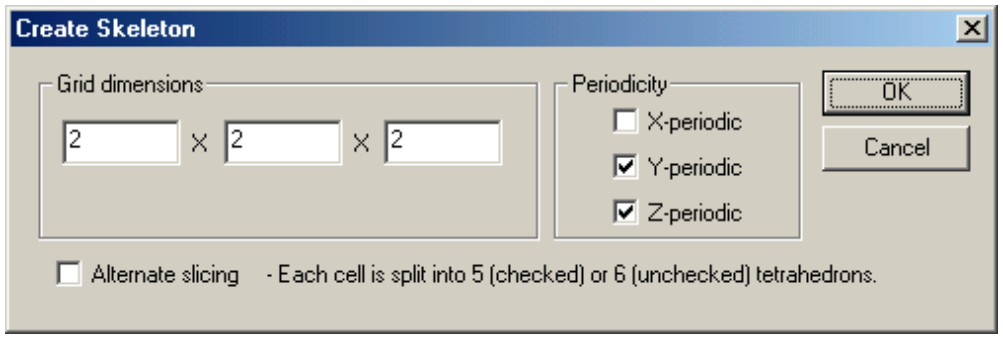
Thank you!

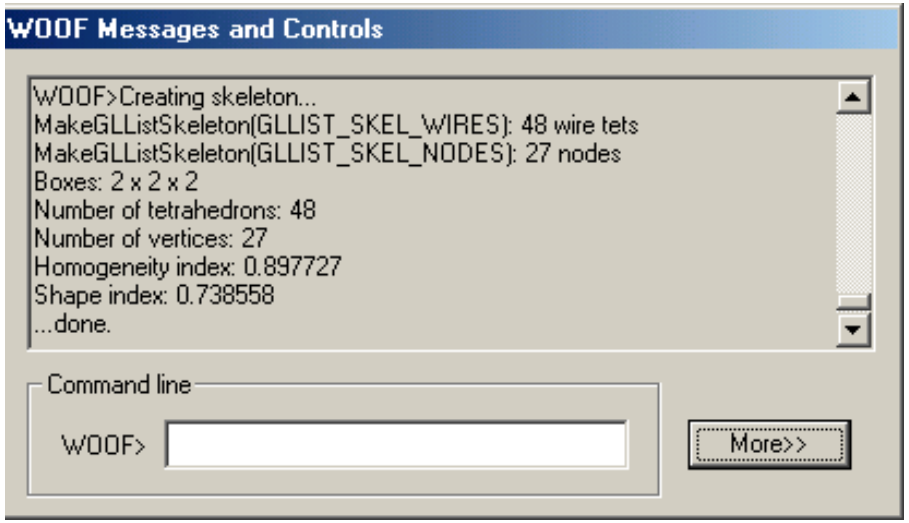
Supplementary material

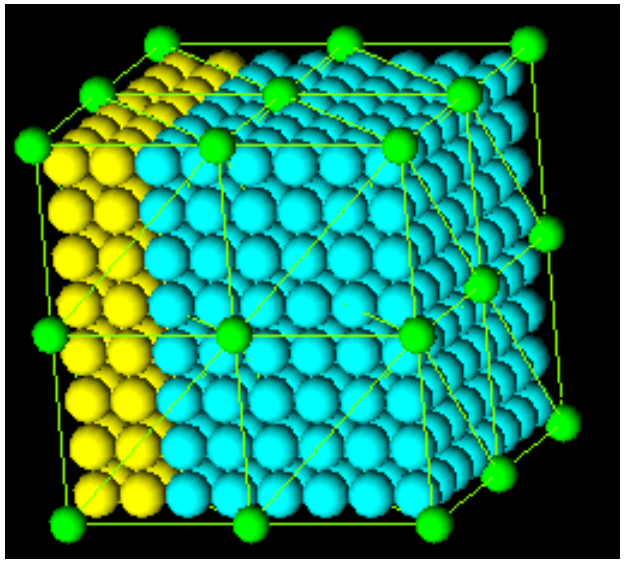
Skeleton-modification using WOOF for a simple microstructure (cyallow3D)



Create an initial skeleton







Display skeleton wires and nodes (green).

Snap-nodes

For the microstructure, display only the yellow pixels to see more clearly what went on…